PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-236584
(43)Date of publication of application : 23.08.2002

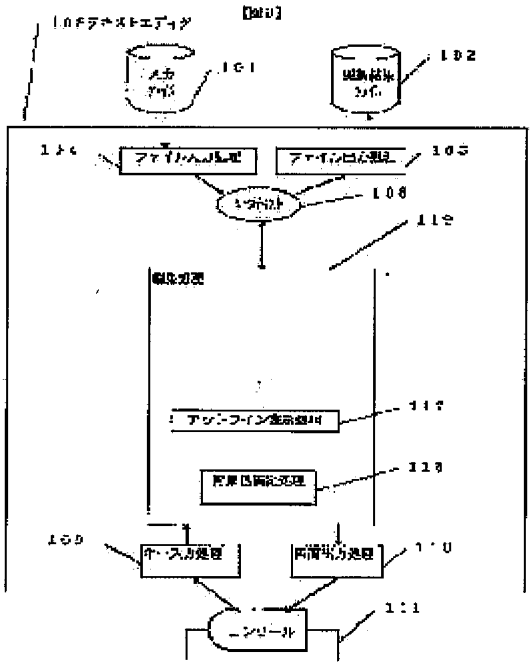
(51)Int.Cl. G06F 9/44

(21)Application number : 2001-031703
(22)Date of filing : 08.02.2001

(71)Applicant : HITACHI LTD
(72)Inventor : SAKANISHI SENICHIRO
KAWAI MASAO

(54) TEXT DISPLAY PROGRAM

(57)Abstract:
PROBLEM TO BE SOLVED: To efficiently display a source text which is developed according to a translation condition indicator as to a text editor for editing the source text including the condition translation indicator.
SOLUTION: A part in the source text which is not affected by a condition translation variable and a part which is affected by the condition translation variable are extracted from the source text respectively and for the extracted source text parts, different background colors are set. According to the depth of the nesting of condition translation indicators, indent displays are made by influence ranges of a translation conditional expression. The condition translation indicators are displayed as display items without fail and the ranges that the condition translation indicators affect are dynamically displayed or not displayed.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2002－236584
（P2002－236584A）
(43)公開日　平成14年８月23日(2002.8.23)

(51)Int.Cl.[7]	識別記号	ＦＩ	テーマコード（参考）
Ｇ０６Ｆ　9/44		Ｇ０６Ｆ　9/06	６２０Ｂ　５Ｂ０７６

審査請求　未請求　請求項の数５　ＯＬ　（全 21 頁）

(21)出願番号　特願2001－31703(P2001－31703)

(22)出願日　平成13年２月８日(2001.2.8)

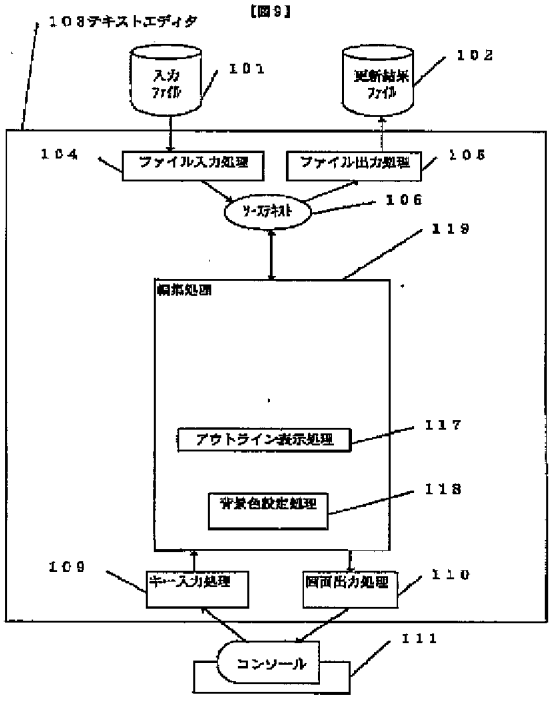
(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　坂西　宣一郎
神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウェア事業部内
(72)発明者　河井　政雄
神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウェア事業部内
(74)代理人　100075096
弁理士　作田　康夫

Ｆターム(参考)　5B076 AB15 DB04 DB05 DB06 EA02 EC02

(54)【発明の名称】　テキスト表示プログラム

(57)【要約】
【課題】条件翻訳指示子を含むソーステキストを編集するテキストエディタに関し、翻訳条件指示子に従って展開されたソーステキストを効率よく表示する。
【解決手段】ソーステキスト中の条件翻訳変数に影響されない部分と、条件翻訳変数に影響される部分とを、それぞれソーステキストから抽出し、抽出した各ソーステキスト部分をそれぞれ別々の色で背景色を設定する処理する。さらに条件翻訳指示子の入れ子の深さに応じて、前記翻訳条件式の影響範囲ごとにインデント表示をおこなう。また、条件翻訳指示子を表示項目として必ず表示し、前記条件翻訳指示子の影響する範囲を動的に表示・非表示する。

【図９】



1

【特許請求の範囲】

【請求項１】条件翻訳指示子を含むプログラムのソーステキストを表示するテキスト表示プログラムにおいて、ソーステキスト中の条件翻訳指示子の翻訳条件式ごとに影響範囲を認識し、ソーステキストを表示する際に、前記翻訳条件式の影響範囲と前記翻訳条件式の影響しない範囲で表示色の異なる文字表示をおこなうことを特徴とするテキスト表示プログラム。

【請求項２】請求項１において、さらに条件翻訳指示子の入れ子の深さに応じて、前記翻訳条件式の影響範囲ごとにインデント表示をおこなうことを特徴とするテキスト表示プログラム。

【請求項３】条件翻訳指示子を含むプログラムのソーステキストを表示するテキスト表示プログラムにおいて、ソーステキスト中の条件翻訳指示子ごとに影響範囲を抽出し、前記影響範囲の条件翻訳指示子を含む行を見出し項目として表示し、前記影響範囲の条件翻訳指示子を除く行の表示・非表示を動的におこなうことを特徴とするテキスト表示プログラム。

【請求項４】条件翻訳指示子の翻訳条件式を満たす範囲にあっては、前記影響範囲の条件翻訳指示子を除く行の表示をおこなうことを特徴とする請求項３に記載のテキスト表示プログラム。

【請求項５】条件翻訳指示子を含むプログラムのソーステキストを表示するテキスト表示プログラムにおいて、ソーステキストから条件翻訳指示子の翻訳変数と設定値を条件翻訳変数テーブルに抽出し、前記条件翻訳変数テーブルの内容の変更に応じて翻訳条件の再判定を行い、ソーステキストの翻訳条件を満たす範囲の再表示をおこないうことを特徴とする請求項３または請求項４記載のテキスト表示プログラム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、プログラム言語のソーステキストファイルを表示あるいは編集するテキストエディタに関し、とりわけ条件翻訳指示子を含むプログラムのソーステキストを表示するのに好適なテキスト表示プログラムに関する。

【０００２】

【従来の技術】プログラム言語には、コンパイラがソーステキスト上のどの部分を読み込むかをコンパイラに指示する条件翻訳指示子が規定されている場合が多い。例えばＣ言語では、#if条件翻訳指示子、#else条件翻訳指示子、#endif条件翻訳指示子が規定されていて、#if条件翻訳指示子に指定された翻訳条件が成立する場合、コンパイラは#if条件翻訳指示子以降#else条件翻訳指示子までの範囲のソーステキストを読み込み翻訳する。また#if条件翻訳指示子に指定された翻訳条件が成立しない場合、コンパイラは#else条件翻訳指示子以降#endif条件翻訳指示子までの範囲のソーステキストを読み込み翻訳する。さらにＣ言語では条件翻訳変数を定義する#define条件翻訳指示子が規定されていて、#if条件翻訳指示子の翻訳条件中でこの条件翻訳変数を使用することもできる。図20に、以降の本発明の説明において使用する条件翻訳指示子の文法と意味をまとめて示す。

2

【０００３】また、以降の本発明の説明において、互いに背反関係にある（ａ）#if条件翻訳指示子から#else条件翻訳指示子で囲まれた範囲のソーステキスト部分、（ｂ）#else条件翻訳指示子から#endif条件翻訳指示子で囲まれた範囲のソーステキスト部分を、翻訳条件に応じてそれぞれ翻訳条件が成立する条件翻訳領域、又は翻訳条件が成立しない条件翻訳領域と表現して区別する。

【０００４】従来は、第２９９０７０１号特許に記載のように、条件翻訳指示子が含まれるソーステキストファイルを読み込む場合、入力ファイル読み込み時に翻訳条件が成立する条件翻訳領域を更新ワークファイルに出力することで、翻訳条件が成立する条件翻訳領域だけを画面に表示するようになっていた。このため、第２９９０７０１号特許では、条件翻訳領域が入れ子構造になっているような複雑な場合については、処理ができないか、もしくは処理できても何度も繰り返し操作する必要があり効率が悪いという問題があった。

【０００５】また特開平５－８８８７６号公報に記載のように、条件翻訳指示子が含まれるソーステキストファイルを読み込む場合、入力ファイルを本文、コメント部、条件翻訳部に区別し、条件翻訳部の翻訳条件を判定し、判定結果に従って条件翻訳部を表示するか否かを決定するようになっていた。しかしこの特開平５－８８８７６号公報では、#else条件翻訳指示子を指定した場合の処理方法について配慮されていないため、#else条件翻訳指示子を含むソーステキストや入れ子構造の条件翻訳領域を含む複雑なソーステキストを効率的に処理することができないという問題があった。

【０００６】更に特開平６－８３６０２号公報に記載のように、条件翻訳指示子が含まれるソーステキストファイルを読み込む場合、入力ファイル読み込み時に、プログラムの実行に関与するソーステキスト部分と、プログラムの実行に関与しないソーステキスト部分に分割してバッファに格納し、プログラムの実行に関与するソーステキスト部分のみを画面に表示するようになっていた。しかしこの特開平６－８３６０２号公報では、プログラムの実行に関与する部分、即ち条件翻訳変数に影響されない共通なソーステキスト部分および翻訳条件が成立する条件翻訳領域において、各領域の境界が不明確で条件翻訳変数に影響されない共通なソーステキスト部分が容易に識別できないため、例えば図２に示すように、共通なソーステキスト部分と非共通なソーステキスト部分を同一部分であると誤認することによる修正ミスが発生しやすいという問題があった。

【０００７】図２において、この従来技術によるテキス

トエディタで入力ファイル１０を読み込んだ場合の表示内容が１１である。ここでテキストエディタ使用者は、条件翻訳変数に影響されない共通なソーステキスト部分と翻訳条件が成立する条件翻訳領域の境界が不明確なため、式１５が条件翻訳変数に影響されない共通なソーステキスト部分であり、式１６が翻訳条件が成立する条件翻訳領域であることを認識できず、式１５と式１６は同一部分であり冗長であると誤って判断してしまい、式１５を変更し式１６を削除して、最終的に１２に示す内容に修正してしまう。ここで編集後のソーステキストをファイルに出力した場合の内容が１３である。１３では、条件翻訳変数に影響されない共通なソーステキスト部分であり本来変更してはならない式１５が式１７に修正されている。その結果、条件翻訳指示子「#if PC == 1」が成立しない場合、変数「a」の値が１０と１３では異なるため、誤動作する可能性がある。

【０００８】

【発明が解決しようとする課題】以上で説明した従来技術では（1）条件翻訳領域が入れ子構造になっているような複雑な条件翻訳があるソーステキストの場合において、翻訳条件に従って展開されたソーステキストを効率よく表示することができない、（2）条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中で条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できないため、共通なソーステキスト部分に対する修正ミスが発生しやすい、という問題があった。

【０００９】本発明の目的は、上記従来技術の問題点に対して、条件翻訳領域が入れ子構造になっているような複雑な条件翻訳があるプログラムのソーステキストの場合において、翻訳条件に従って展開されたソーステキストを効率よく表示するテキストエディタを提供することにあり、また、条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中で条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できるようにし、共通なソーステキスト部分に対する修正ミスが起りにくいテキストエディタを提供することにある。

【００１０】

【課題を解決するための手段】上記目的は、ソーステキスト中から各条件翻訳領域を抽出し、条件翻訳指示子の翻訳条件式ごとに影響範囲を認識し、ソーステキストを表示する際に、前記翻訳条件式に影響範囲と前記翻訳条件式に影響されない範囲を視覚的に異なる表示をおこうようにすることで達成される。例えば、前記範囲のテキストを異なる色分け表示をおこなうようにする。

【００１１】さらに、前記翻訳条件式の影響範囲が、複数の条件翻訳領域の入れ子構造になっている場合には、入れ子の深さに応じて翻訳条件式の影響範囲をインデント表示するようにする。より詳細には、複数の翻訳条件式の影響範囲が入れ子の構造になっている場合、入れ子になっている内側の翻訳条件の影響範囲のソーステキストと該影響範囲の外側のソーステキストをインデント表示をするとともに異なる表示色でテキスト表示するようにする。

【００１２】さらに、上記条件翻訳領域の表示をおこなう際に、翻訳条件式が成立する条件翻訳領域のみ表示し、翻訳条件式が成立しない条件翻訳領域は表示しないようにしてもよい。

【００１３】また、条件翻訳指示子を境界としてソーステキストを複数の領域に分割するとともに、各領域の境界に相当する条件翻訳指示子は必ず表示し、分割後の各領域を、条件翻訳が成立するか否かという条件、もしくはテキストエディタ使用者の指示に従って動的に表示・非表示を制御するようにする。

【００１４】さらに、ソーステキストから条件翻訳指示子の翻訳変数と設定値を読み出し、プログラムの条件翻訳変数テーブルに格納し、該条件翻訳テーブルを表示・変更可能とし、前記テーブルに基づき、上記のとおりソーステキストを表示するようにする。

【００１５】

【発明の実施の形態】以下、本発明によるテキストエディタについて図をもちいて詳細に説明する。

【００１６】（実施例１）本実施例では、ソーステキスト中の条件翻訳指示子の翻訳条件を判定し、翻訳条件が成立する条件翻訳領域を条件翻訳指示子に対応したそれぞれ別々の色で背景色を色分けし、かつ条件翻訳領域の入れ子の深さに応じてインデント表示し、翻訳条件が成立しない条件翻訳領域を表示しないことによって、翻訳条件に従って展開されたソーステキストを効率よく表示するとともに、ソーステキスト中の条件翻訳変数に影響されない共通なソーステキスト部分をテキストエディタ使用者が容易に識別できるように、前述の条件翻訳領域と区別可能な色で背景色を色分け表示する。更に、各条件翻訳変数に影響されるソーステキスト部分において、互いに背反関係にある、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方で共通なソーステキスト部分を、テキストエディタ使用者が容易に識別できるように、前述の条件翻訳領域他と区別可能な色で背景色を色分け表示する。

【００１７】なお、以下の説明において、条件翻訳変数に影響されない共通なソーステキスト部分を共通ソーステキスト部分１と表現し、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方で共通なソーステキスト部分を共通ソーステキスト部分２と表現して区別する。

【００１８】図１は、本発明の第一の実施例を実現するテキストエディタの構成例を示すブロック図である。図１において、テキストエディタ１０３は、プログラム言語のソーステキストを内容とする入力ファイル１０１を

外部記憶装置から読み込みソーステキスト１０６としてテキストエディタ１０３内部に保持するファイル入力処理１０４と、コンソール１１１からのテキストエディタ使用者の操作を入力するキー入力手段１０９と、前記操作に基づいてソーステキスト１０６を編集する編集処理１０７と、この編集結果によって更新されたソーステキスト１０６をコンソール１１１に表示する画面出力処理１１０と、前記編集結果によって更新されたソーステキスト１０６を外部記憶装置の更新結果ファイル１０２に出力するファイル出力処理１０５から構成される。

【００１９】編集処理１０７は、さらに、ソーステキスト１０６中の条件翻訳領域に対して、翻訳条件が成立する条件翻訳領域をコンソール１１１に表示する条件成立翻訳領域表示処理１１３と、ソーステキスト１０６中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定する背景色設定処理１１８から構成される。

【００２０】図３は、以下の説明において、本発明によるテキストエディタが読み込む入力ファイルの例である。

【００２１】図４は、図３のソーステキストを図１の構成から成るテキストエディタで読み込んだ場合の、コンソール１１１の表示例である。

【００２２】図４において、コンソール１１１の表示画面には、入力ファイル１０１から読み込んだソーステキスト１０６を表示するソーステキストウィンドウ３０１を表示し、テキストエディタ使用者は、このソーステキストウィンドウ３０１に対して編集作業を行う。

【００２３】図５は、図４のコンソール１１１に表示されているソーステキストウィンドウ３０１の表示内容の詳細例である。

【００２４】図５では、ソーステキストウィンドウ３０１に、条件翻訳変数に影響されない共通ソーステキスト部分１と、翻訳条件が成立する条件翻訳領域を表示する。

【００２５】図５において、４０１はソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分１であり、テキストエディタ使用者が共通ソーステキスト部分１を容易に識別できるように、背景色設定処理１１８により指定された色で背景色を色分け表示する。４０２は、各条件翻訳変数に影響されるソーステキスト部分において、互いに背反関係にある、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方で共通な共通ソーステキスト部分２であり、テキストエディタ使用者が共通ソーステキスト部分２を容易に識別できるように、背景色設定処理１１８により指定された色で背景色を色分け表示する。４０３は条件翻訳変数「ＰＣ」に影響されるソーステキスト部分であり、４０４は条件翻訳変数「ＷＳ」に影響されるソーステキスト部分であり、４０５は条件翻訳変数「ＭＡＣＨＩＮＥ１」に影響されるソーステキスト部分であり、テキストエディタ使用者が条件翻訳変数に影響されない共通ソーステキスト部分１である４０１と容易に区別できるように、かつそれぞれの条件翻訳変数に影響されるソーステキスト部分を互いに区別できるように、背景色設定処理１１８によりそれぞれ指定された別々の色で背景色を色分け表示するとともに、条件翻訳指示子の入れ子の深さに応じてインデント表示する。

【００２６】図３に示した入力ファイル例では、条件翻訳変数「ＭＡＣＨＩＮＥ１」に影響されるソーステキスト部分は、条件翻訳変数「ＷＳ」に影響されるソーステキスト部分の入れ子となっており、更にこの条件翻訳変数「ＷＳ」に影響されるソーステキスト部分は、条件翻訳変数「ＰＣ」に影響されるソーステキスト部分の入れ子となる構造となっている。このように条件翻訳変数に影響される部分が入れ子構造になっているソーステキストの場合、図５に示すように本実施例では、入れ子の深さに応じてインデント表示するとともに、各条件翻訳変数毎に背景の色を定め、更に入れ子の深さが一番深い部分が一番手前になるよう重ねて表示する。この結果、条件翻訳領域に影響されるソーステキスト部分が入れ子構造になっていた場合でも、どの条件翻訳変数に影響を受けるのかを背景色で明示しながらソーステキストを表示することができる。

【００２７】図５において、４０６、４０７および４０８は、各条件翻訳変数に影響されるソーステキスト部分のインデント領域であり、条件翻訳変数に影響されるソーステキスト部分が入れ子構造になっている場合、入れ子階層を明確化するとともに、テキストエディタ使用者がソーステキストがどの入れ子階層に属するか容易に識別できるように、各条件翻訳変数毎に定めた背景色で塗りつぶす。

【００２８】なお図５では、４０１、４０２，４０３、４０４及び４０５を枠で囲んで表現しているが、それぞれテキストエディタ使用者によって指示された色で背景色を表示するだけでもよい。更に図５では、４０１，４０２，４０３，４０４及び４０５の背景色を色分け表示しているが、背景色ではなく文字色を色分け表示してもよい。

【００２９】また図５では、各条件翻訳領域をインデントして表示しているが、条件翻訳領域の入れ子が深くなった場合にソーステキストの連続性が失われるという点に配慮し、テキストエディタ使用者からの指示によりソーステキストの文法に従ったインデントで表示することもできる。

【００３０】次に、本実施例によるテキストエディタの処理手順をフローチャートを用いて説明する。

【００３１】図６は、図１の構成から成るテキストエディタの処理の流れを説明するフローチャートである。

【００３２】図６では、入力ファイル１０１からソーステキスト１０６を読み込み（ステップ５０１）、ソーステキスト１０６中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定し（ステップ５０２。本処理の詳細は後述する図７で説明する。なおフローチャート中で縦線が二重になっている箱は、別の処理を呼び出すことを示す）、条件翻訳変数に影響されない共通ソーステキスト部分１と翻訳条件が成立する条件翻訳領域をコンソール１１１に表示する（ステップ５０３。本処理の詳細は後述する図８で説明する）。

【００３３】次にテキストエディタ使用者からの編集操作を受け取ったキー入力処理１０９からの指示に従い、ソーステキスト１０６に対して編集を行い（ステップ５０４）、編集操作によって更新されたソーステキスト１０６をコンソール１１１に表示する（ステップ５０５）。

【００３４】ここでテキストエディタ使用者がテキストエディタ終了を指示したか判定し（ステップ５０６）、終了を指示した場合はソーステキスト１０６を更新結果ファイル１０２に出力し（ステップ５０７）、テキストエディタを終了する。

【００３５】テキストエディタ終了が指示されていない場合、テキストエディタ使用者が#define条件翻訳指示子の変更または新規追加を行ったか判定し（ステップ５０８）、#define条件翻訳指示子の変更または新規追加を行っていない場合、ステップ５０４に戻り以降のステップを繰り返す。

【００３６】テキストエディタ使用者が#define条件翻訳指示子の変更または新規追加を行った場合、ステップ５０２に戻り、共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ５０３で条件翻訳変数に影響されない共通ソーステキスト部分１と翻訳条件が成立する条件翻訳領域をコンソール１１１に再表示する。そしてステップ５０４以降のステップを繰り返す。

【００３７】図７は、図６におけるステップ５０２で実行される、背景色設定処理１１８の処理の流れを説明するフローチャートである。

【００３８】図７では、ソーステキスト１０６からソーステキストを１行取り出し（ステップ７０１）、ソーステキスト終端か判定し（ステップ７０２）、ソーステキスト終端の場合は呼び出し元に処理を戻す。

【００３９】ソーステキスト終端以外の場合、取り出したソーステキスト行の行頭が「#if」であるか判定し（ステップ７０３）、行頭が「#if」以外の場合は、ソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分１としてテキストエディタ使用者に指定された色で背景色を色分け表示するよう取り出した行に設定し（ステップ７０４）、ステップ７０１に戻り以降のステップを繰り返す。

【００４０】取り出したソーステキスト行の行頭が「#if」の場合、ソーステキスト１０６から行頭が「#else」であるソーステキスト行までの範囲を取り出し退避し（ステップ７０５）、次にソーステキスト行を取り出す位置を「#else」の次の行に設定し（ステップ７０６）、ソーステキスト１０６から行頭が「#endif」であるソーステキスト行までの範囲を取り出し退避し（ステップ７０７）、次にソーステキスト行を取り出す位置を「#endif」の次の行に設定する（ステップ７０８）。

【００４１】次にステップ７０５およびステップ７０７で退避した２つのソーステキストに対して、対応する条件翻訳変数に影響されるソーステキスト部分としてテキストエディタ使用者に指定された色で背景色を色分け表示するよう設定する（ステップ７０９）。

【００４２】次にステップ７０５およびステップ７０７で退避した２つのソーステキストを比較し、双方に共通なソーステキスト部分とそれ以外のソーステキスト部分に分割し（ステップ７１０）、分割した共通なソーステキスト部分に対して、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方に共通な共通ソーステキスト部分２としてテキストエディタ使用者に指定された色で背景色を色分け表示するよう設定する（ステップ７１１）。

【００４３】次にステップ７０５およびステップ７０７で退避した２つのソーステキストに対して本処理を再帰的に実行し（ステップ７１２）、次にソーステキスト行を取り出す位置をステップ７０８で設定した位置に再設定し（ステップ７１３）、ステップ７０１に戻り以降のステップを繰り返す。

【００４４】図８は、図６におけるステップ５０３で実行される、条件成立翻訳領域表示処理１１３の処理の流れを説明するフローチャートである。

【００４５】図８では、ソーステキスト１０６からソーステキストを１行取り出し（ステップ８０１）、ソーステキスト終端か判定し（ステップ８０２）、ソーステキスト終端の場合は呼び出し元に処理を戻す。

【００４６】ソーステキスト終端以外の場合、取り出したソーステキスト行の行頭が「#if」であるか判定し（ステップ８０３）、行頭が「#if」以外の場合は、取り出したソーステキスト行をコンソール１１１に表示するよう画面出力処理１１０に指示し（ステップ８０４）、ステップ８０１に戻り以降のステップを繰り返す。

【００４７】取り出したソーステキスト行の行頭が「#if」の場合、#if条件翻訳指示子から翻訳条件を抽出し（ステップ８０５）、ソーステキスト１０６から行頭が「#else」であるソーステキスト行までの範囲を取り出し退避し（ステップ８０６）、次にソーステキスト行を

取り出す位置を「#else」の次の行に設定し（ステップ８０７）、ソーステキスト１０６から行頭が「#endif」であるソーステキスト行までの範囲を取り出し退避し（ステップ８０８）、次にソーステキスト行を取り出す位置を「#endif」の次の行に設定する（ステップ８０９）。

【００４８】次にステップ８０５で抽出した翻訳条件を判定する（ステップ８１０）。翻訳条件が成立する場合、ステップ８０６で退避した、#if条件翻訳指示子から#else条件翻訳指示子までの範囲のソーステキストをコンソール１１１に条件翻訳領域の入れ子の深さに応じてインデントして表示するよう画面出力処理１１０に指示する（ステップ８１１）。ステップ８０５で抽出した翻訳条件が成立しない場合、ステップ８０８で退避した、#else条件翻訳指示子から#endif条件翻訳指示子までの範囲のソーステキストをコンソール１１１に条件翻訳領域の入れ子の深さに応じてインデントして表示するよう画面出力処理１１０に指示する（ステップ８１２）。

【００４９】次にステップ８０６及びステップ８０８で退避したソーステキストに対して本処理を再帰的に実行し（ステップ８１３）、次にソーステキスト行を取り出す位置をステップ８０９で設定した位置に再設定し（ステップ８１４）、ステップ８０１に戻り以降のステップを繰り返す。

【００５０】なお図８の各ステップにおいて画面出力を行う場合は、図６のステップ５０２によって設定された色で背景色を色分け表示する。

【００５１】以上で説明した実施例によって、従来技術で問題であった、条件翻訳領域が入れ子構造になっているような複雑な条件翻訳を含むソーステキストの場合に、翻訳条件に従って展開されたソーステキストを効率よく表示することができないという問題を軽減することができる。また、従来技術で問題であった、条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中で条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できないため、共通なソーステキスト部分に対する修正ミスが発生しやすいという問題を軽減することもできる。更に本実施例によれば、ソーステキスト中の翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方に共通なソーステキスト部分が存在する場合、このソーステキスト部分は翻訳条件の判定結果にかかわらず常に実行されるため、条件翻訳領域内に存在する必要はなく条件翻訳領域の外に移動すべきであることを、テキストエディタ使用者が容易に判断できるという効果を持つ。

【００５２】なお、以上で説明した実施例では、ソーステキスト中の条件翻訳指示子の翻訳条件を判定し、翻訳条件が成立する条件翻訳領域だけを表示するという表示方法であるが、本実施例における条件翻訳指示子の翻訳条件の判定および判定結果に基づいて条件翻訳領域の表示の有無を決定する処理を実行せず、更には条件翻訳領域以下をインデントして全て表示することによって、ソーステキスト全てを表示することができ、かつソーステキスト中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響される各ソーステキスト部分をそれぞれ別々の色で背景色を色分け表示するようにできることは言うまでもない。この方法と本実施例で説明した方法を選択するように構成することによって、テキストエディタ使用者は、翻訳条件が成立する条件翻訳領域だけを表示するかソーステキスト全てを表示するか、必要に応じて切替えることができる。

【００５３】本実施例で、ソーステキストの区別を背景色の変更によりおこなう例を説明したが、文字色の変更や、文字種別により区別してもよいことは言うまでもない。

【００５４】（実施例２）次に、本発明の第二の実施例によるテキストエディタについて説明する。

【００５５】本発明の第二の実施例は、第一の実施例とは別の手段によって従来技術の問題点を解決することを目的とした実施例である。

【００５６】本実施例では、条件翻訳指示子を境界としてソーステキストを複数の領域に分割するとともに、各領域の境界に相当する条件翻訳指示子は必ず表示し、分割後の各領域を、テキストエディタ使用者の指示に従って動的に表示・非表示を制御する。本実施例ではこれをアウトライン表示と表現し、前記の分割した領域をアウトライン表示領域と表現する。

【００５７】図９は、本発明の第二の実施例を実現するテキストエディタの構成例を示すブロック図である。

【００５８】図９において、編集処理１１９を除いて、その構成は図１の実施例１の構成と同様であり、同一のブロックは同一の記号で示している。

【００５９】図９における編集処理１１９は、各条件翻訳領域のアウトライン表示を行うアウトライン表示処理１１７と、ソーステキスト１０６中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定する背景色設定処理１１８から構成される。本実施例の場合、図３のソーステキストを図９の構成から成るテキストエディタで読み込んだ場合のコンソール１１１の表示方法は、図４に示す本発明の第一の実施例の場合と同様であるため、説明は省略する。

【００６０】図１０は、図３のソーステキストを図９の構成から成るテキストエディタで読み込んだ場合の、ソーステキストウィンドウ３０１の表示例の詳細である。

【００６１】本実施例の中核部分は、#if、#elseおよび#endifの各条件翻訳指示子を、入れ子の深さに応じてインデント処理して全て表示するとともに、これら条件翻訳指示子によって囲まれたソーステキストについては、

各条件翻訳指示子毎にアウトライン表示領域展開状態指示子を設け、その指示に従ってソーステキストを表示するか否かの制御をすることにある。ここで、アウトライン表示領域展開状態指示子への設定は、#if条件翻訳指示子の翻訳条件が成立する条件翻訳領域は表示をし、逆に翻訳条件が成立しない条件翻訳領域は表示をしないという状態を初期状態とし、以降はテキストエディタ使用者の指示によって表示・非表示を制御するという方法である。

【００６２】図１０において、４０１はソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分１であり、テキストエディタ使用者が共通ソーステキスト部分１を容易に識別できるように、背景色設定処理１１８により指定された色で背景色を色分け表示する。４０２は、各条件翻訳変数に影響されるソーステキスト部分において、互いに背反関係にある、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方で共通な共通ソーステキスト部分２であり、テキストエディタ使用者が共通ソーステキスト部分２を容易に識別できるように、背景色設定処理１１８により指定された色で背景色を色分け表示する。４０９は条件翻訳指示子「#if PC == 1」に対応する#else条件翻訳指示子と#endif条件翻訳指示子を境界とする展開状態のアウトライン表示領域であり、条件翻訳領域が入れ子構造になっている場合、このアウトライン表示領域の中に、入れ子の条件翻訳領域に対応するアウトライン表示領域がインデント処理されて表示される。４１０は条件翻訳指示子「#if WS == 1」と#else条件翻訳指示子を境界とする展開状態のアウトライン表示領域であり、アウトライン表示領域４０９の中に包含されている。４１１は条件翻訳指示子「#if MACHINE1 == 1」と#else条件翻訳指示子を境界とする展開状態のアウトライン表示領域であり、アウトライン表示領域４１０の中に包含されている。４１２は展開されたアウトライン表示領域を記号「－」で表現し、展開されていないアウトライン表示領域を記号「＋」で表現するアウトライン表示領域展開状態指示子であり、テキストエディタ使用者は、このアウトライン表示領域展開状態指示子４１２に対して操作を行うことにより、アウトライン表示領域の展開・非展開状態を変更することができる。

【００６３】なお本実施例では、テキストエディタの起動直後の各アウトライン表示領域の展開・非展開状態は、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域は展開状態で、翻訳条件が成立しない条件翻訳領域に対応するアウトライン表示領域は非展開状態となるように設定する。図１０はこのような設定で表示した例であり、翻訳条件が成立する条件翻訳領域に対応する、条件翻訳指示子「#if PC == 1」に対応する#else条件翻訳指示子と#endif条件翻訳指示子を境界とするアウトライン表示領域４０９と、条件翻訳指示子「#if WS == 1」と#else条件翻訳指示子を境界とするアウトライン表示領域４１０と、条件翻訳指示子「#if MACHINE1 == 1」と#else条件翻訳指示子を境界とするアウトライン表示領域４１１を展開状態として表示し、その他の翻訳条件が成立しない条件翻訳領域に対応するアウトライン表示領域はすべて非展開状態で表示している。

【００６４】また本実施例では、各アウトライン表示領域の展開・非展開状態をテキストエディタ使用者が変更しても、テキストエディタ起動直後の状態である、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域だけを展開状態にするように戻すこともできる。

【００６５】なお図１０では、４０１及び４０２を枠で囲んで表現しているが、テキストエディタ使用者によって指示された色で背景色を表示するだけでもよい。更に図１０では、４０１及び４０２の背景色を色分け表示しているが、背景色ではなく文字の色を色分け表示してもよい。

【００６６】図１１は、図１０における条件翻訳指示子「#if PC == 1」と#else条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を、例えば、「#if PC ==1」の行頭にあるアウトライン表示領域展開状態指示子４１２をテキストエディタ使用者がマウスでダブルクリックするというような操作を行うことで、展開させた場合の表示例である。

【００６７】図１１において、条件翻訳指示子「#if PC == 1」の行頭にあるアウトライン表示領域展開状態指示子４１２の記号が、非展開状態を示す「＋」から、展開状態を示す「－」に変更されているとともに、図１０では表示されていなかったアウトライン表示領域４１３が展開状態で表示されている。

【００６８】以下、本実施例によるテキストエディタの処理手順をフローチャートを用いて説明する。

【００６９】図１２は、図９の構成から成るテキストエディタの処理の流れを説明するフローチャートである。

【００７０】図１２では、入力ファイル１０１からソーステキスト１０６を読み込み（ステップ１４０１）、ソーステキスト１０６中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定し（ステップ１４０２。なお本処理は実施例１の図６におけるステップ５０２と同様であり、詳細は図７を参照）、ソーステキスト１０６をコンソール１１１にアウトライン表示する（ステップ１４０３。本処理の詳細は後述する図１３で説明する）。

【００７１】次にテキストエディタ使用者からの編集操作を受け取ったキー入力処理１０９からの指示に従い、ソーステキスト１０６に対して編集を行い（ステップ１４０４）、編集操作によって更新されたソーステキスト１０６をコンソール１１１に表示する（ステップ１４０５）。

【００７２】ここでテキストエディタ使用者がテキストエディタ終了を指示したか判定し（ステップ１４０６）、終了を指示した場合はソーステキスト１０６を更新結果ファイル１０２に出力し（ステップ１４０７）、テキストエディタを終了する。

【００７３】テキストエディタ終了が指示されていない場合、テキストエディタ使用者が任意のアウトライン表示領域に対して展開・非展開状態の変更操作を行ったか判定し（ステップ１４０８）、変更操作を行った場合、対応するアウトライン表示領域の展開・非展開状態を変更し、変更したアウトライン表示領域をコンソール１１１に再表示し（ステップ１４０９）、ステップ１４０４に戻り以降のステップを繰り返す。

【００７４】テキストエディタ使用者がアウトライン表示領域の展開・非展開状態の変更操作を行っていない場合、テキストエディタ使用者が#define条件翻訳指示子の変更または新規追加を行ったか判定し（ステップ１４１０）、変更または新規追加を行った場合、ステップ１４０２に戻り、共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ１４０３以降のステップを繰り返す。

【００７５】テキストエディタ使用者が#define条件翻訳指示子の変更または新規追加を行っていない場合、テキストエディタ使用者が、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域だけを展開表示するよう指示したか判定し（ステップ１４１１）、指示していない場合、ステップ１４０４に戻り以降のステップを繰り返す。

【００７６】テキストエディタ使用者が、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域だけを展開表示するよう指示した場合、ステップ１４０３に戻り、全てのアウトライン表示領域に対して、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域を展開状態に設定し、翻訳条件が成立しない条件翻訳領域に対応するアウトライン表示領域を非展開状態に設定し、全てのアウトライン表示領域をコンソール１１１に再表示し、ステップ１４０４以降のステップを繰り返す。

【００７７】図１３は、図１２におけるステップ１４０３で実行される、アウトライン表示処理１１７の処理の流れを説明するフローチャートである。

【００７８】図１３では、ソーステキスト１０６からソーステキストを１行取り出し（ステップ１５０１）、ソーステキスト終端か判定し（ステップ１５０２）、ソーステキスト終端の場合は呼び出し元に処理を戻す。

【００７９】ソーステキスト終端以外の場合、取り出したソーステキスト行の行頭が「#if」であるか判定し（ステップ１５０３）、行頭が「#if」以外の場合は、取り出したソーステキスト行をコンソール１１１に表示するよう画面出力処理１１０に指示し（ステップ１５０４）、ステップ１５０１に戻り以降のステップを繰り返す。

【００８０】取り出したソーステキスト行の行頭が「#if」の場合、#if条件翻訳指示子から翻訳条件を抽出し（ステップ１５０５）、ソーステキスト１０６から行頭が「#else」であるソーステキスト行までの範囲を取り出し退避し、#if条件翻訳指示子と#else条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を作成し（ステップ１５０６）、次にソーステキスト行を取り出す位置を「#else」の次の行に設定し（ステップ１５０７）、ソーステキスト１０６から行頭が「#endif」であるソーステキスト行までの範囲を取り出し退避し、#else条件翻訳指示子と#endif条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を作成し（ステップ１５０８）、次にソーステキスト行を取り出す位置を「#endif」の次の行に設定する（ステップ１５０９）。

【００８１】次にステップ１５０５で抽出した翻訳条件を判定する（ステップ１５１０）。翻訳条件が成立する場合、ステップ１５０６で作成した、#if条件翻訳指示子と#else条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を展開状態に設定する（ステップ１５１１）。翻訳条件が成立しない場合、ステップ１５０８で作成した、#else条件翻訳指示子と#endif条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を展開状態に設定する（ステップ１５１２）。

【００８２】次にステップ１５０６およびステップ１５０８で作成した２つのアウトライン表示領域をコンソール１１１に条件翻訳領域の入れ子の深さに応じてインデントして表示するよう画面出力処理１１０に指示し（ステップ１５１３）、ステップ１５０６およびステップ１５０８で退避したソーステキストに対して本処理を再帰的に実行し（ステップ１５１４）、次にソーステキスト行を取り出す位置をステップ１５０９で設定した位置に再設定し（ステップ１５１５），ステップ１５０１に戻り以降の処理を繰り返す。

【００８３】なお図１３の各ステップにおいて画面出力を行う場合は、図１２のステップ１４０２によって設定された色で背景色を色分け表示する。

【００８４】以上で説明した実施例によって、従来技術で問題であった、条件翻訳領域が入れ子構造になっているような複雑な条件翻訳を含むソーステキストの場合に、翻訳条件に従って展開されたソーステキストを効率よく表示することができないという問題を軽減することができる。また、従来技術で問題であった、条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中で条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できないため、共通なソーステキスト部分に対する修正ミスが発生しやすいという問題を軽減することもできる。更に本実施例によれば、

条件翻訳領域をテキストエディタ使用者の指示に従って動的に表示・非表示を制御することができるという効果と、ソーステキスト中の翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方に共通なソーステキスト部分が存在する場合、このソーステキスト部分は翻訳条件の判定結果にかかわらず常に実行されるため、条件翻訳領域内に存在する必要はなく条件翻訳領域の外に移動すべきであることを、テキストエディタ使用者が容易に判断できるという効果を持つ。

【００８５】なお、以上で説明した実施例では、ソーステキスト中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を色分け表示するという表示方法であるが、本実施例における背景色設定処理を実行しないことによって、それぞれを色分け表示せずに、各条件翻訳領域のアウトライン表示だけにすることもできる。このような方法においても、条件翻訳変数に影響されない共通ソーステキスト部分１と各条件翻訳変数に影響されるソーステキスト部分のそれぞれがアウトライン表示によって明確化されるため、従来技術の問題点を解決することができる。

【００８６】（実施例３）次に、本発明の第三の実施例によるテキストエディタについて説明する。

【００８７】本発明の第三の実施例は、本発明の第一の実施例により表示したソーステキストに対して、テキストエディタ使用者が、条件翻訳変数を容易に変更する手段を提供することで、種々の翻訳条件の設定を可能にし、設定された翻訳条件に基づいて展開されたソーステキストを容易に確認できるようにする実施例である。

【００８８】本実施例では、ソーステキスト中の#define条件翻訳指示子から条件翻訳変数の名称と値を抽出した条件翻訳変数テーブルを作成し、作成した条件翻訳変数テーブルの内容をソーステキストを表示するウィンドウとは別のウィンドウに表示し、テキストエディタ使用者がこのウィンドウに表示された条件翻訳変数テーブルの内容を参照・変更すると、この変更内容をソーステキスト中の対応する#define条件翻訳指示子に反映し、条件翻訳指示子の翻訳条件の再判定を行い、翻訳条件を反映したソーステキストを表示する。

【００８９】図１４は、本発明の第三の実施例を実現するテキストエディタの構成例を示すブロック図である。

【００９０】図１４において、編集処理１２０を除いて、その構成は図１の実施例１の構成と同様であり、同一ブロックは同一の記号で示している。

【００９１】図１４における編集処理１２０は、図１の編集処理１０７に対して、ソーステキスト１０６中の条件翻訳変数を抽出し条件翻訳変数テーブル１０８を作成する条件翻訳変数テーブル作成処理１１４と、条件翻訳変数テーブル１０８の内容を表示し、テキストエディタ使用者からの操作を受け取り、ソーステキスト１０６の表示内容と条件翻訳変数テーブル１０８の内容を更新する条件翻訳変数テーブル表示制御処理１１５を追加した構成である。

【００９２】図１５は、図３のソーステキストを図１４の構成から成るテキストエディタで読み込んだ場合のコンソール１１１の表示例である。

【００９３】図１５において、コンソール１１１の表示画面には、入力ファイル１０１から読み込んだソーステキスト１０６を表示するソーステキストウィンドウ３０１と、条件翻訳変数テーブル１０８の内容を表示した条件翻訳変数ウィンドウ３０２を表示する。

【００９４】本実施例の場合、図１５のコンソール１１１に表示されているソーステキストウィンドウ３０１の表示内容の詳細は、図５に示す本発明の第一の実施例の場合と同様であるため、説明は省略する。

【００９５】図１６は、図１５のコンソール１１１に表示されている条件翻訳変数ウィンドウ３０２の表示内容の詳細例である。なお、図１６の内容は、図１４における条件翻訳変数テーブル１０８の内容と同じである。

【００９６】テキストエディタ使用者は、ソーステキストウィンドウ３０１に対して編集作業を行うとともに、条件翻訳変数ウィンドウ３０２に対して条件翻訳変数の名称や値の変更操作を行うことができる。テキストエディタ使用者が条件翻訳変数ウィンドウ３０２に対して変更操作を行った場合、変更操作を条件翻訳変数テーブル１０８に反映するとともに、ソーステキスト１０６中の対応する#define条件翻訳指示子に反映し、ソーステキストウィンドウ３０１の表示内容が更新される。またテキストエディタ使用者がソーステキストウィンドウ３０１に表示されている#define条件翻訳指示子を追加・変更した場合、変更内容を条件翻訳変数テーブル１０８に反映させ、条件翻訳変数ウィンドウ３０２の表示内容を更新する。更に、テキストエディタ使用者が条件翻訳変数ウィンドウ３０２に対して、ソーステキスト中の共通ソーステキスト部分１および共通ソーステキスト部分２、または各条件翻訳変数に影響されるソーステキスト部分の背景色を変更した場合、対応するソーステキスト部分を変更された背景色で再色分け表示する。

【００９７】図１６において、１８０１は条件翻訳変数テーブルの出現順の番号である。１８０２は条件翻訳変数の名称であり、１８０３は条件翻訳変数の値であり、テキストエディタ使用者はそれぞれの内容を変更することができる。１８０４は条件翻訳変数を定義しているソーステキスト行番号である。１８０５はこの条件翻訳変数に影響されるソーステキスト部分の背景色であり、テキストエディタ使用者は内容を変更することができる。

【００９８】また図１６の表中の項番４は、ソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分１を示す特別な項番であり、テキストエディタ使用者は、項番４の項目１８０５を変更することができ、

ソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分１を、変更された背景色で再色分け表示する。同様に図１６の表中の項番５は、ソーステキスト中の共通ソーステキスト部分２を示す特別な項番であり、テキストエディタ使用者は、項番５の項目１８０５を変更することができ、ソーステキスト中の共通ソーステキスト部分２を、変更された背景色で再色分け表示する。

【００９９】以下、本実施例によるテキストエディタの処理手順をフローチャートを用いて説明する。図１７は、図１４の構成から成るテキストエディタの処理の流れを説明するフローチャートである。

【０１００】図１７では、入力ファイル１０１からソーステキスト１０６を読み込み（ステップ１９０１）、ソーステキスト１０６中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定し（ステップ１９０２。なお本処理は実施例１の図６におけるステップ５０２と同様であり、詳細は図７を参照）、条件翻訳変数テーブル１０８を作成し（ステップ１９０３。本処理の詳細は後述する図１８で説明する）、ソーステキスト１０６中の条件翻訳変数に影響されない共通ソーステキスト部分１と、翻訳条件が成立する条件翻訳領域をコンソール１１１に表示し（ステップ１９０４。なお本処理は実施例１の図６におけるステップ５０３と同様であり、詳細は図８を参照）、条件翻訳変数テーブル１０８の内容を条件翻訳変数ウィンドウ３０２に表示する（ステップ１９０５）。

【０１０１】次にテキストエディタ使用者からの条件翻訳変数ウィンドウ３０２に対する操作を待つ（ステップ１９０６）。

【０１０２】テキストエディタ使用者からの操作がテキストエディタ終了を指示しているか判定し（ステップ１９０７）、終了を指示した場合はソーステキスト１０６を更新結果ファイル１０２に出力し（ステップ１９０８）、テキストエディタを終了する。

【０１０３】テキストエディタ終了が指示されていない場合、テキストエディタ使用者が条件翻訳変数ウィンドウ３０２に対して変更操作を行ったか判定し（ステップ１９０９）、変更操作を行った場合、変更内容を対応するソーステキスト１０６中の#define条件翻訳指示子に反映させ（ステップ１９１０）、条件翻訳変数ウィンドウ３０２に指定された背景色で色分け処理するように背景色設定処理１１８に指示し（ステップ１９１１）、ステップ１９０２に戻り、ソーステキスト１０６中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ１９０３で条件翻訳変数テーブルを再構築し、ステップ１９０４でソーステキスト１０６中の条件翻訳変数に影響されない共通ソーステキスト部分１と、翻訳条件が成立する条件翻訳領域をコンソール１１１に再表示し、ステップ１９０５で条件翻訳変数テーブル１０８の内容を条件翻訳変数ウィンドウ３０２に再表示し、以降のステップを繰り返す。

【０１０４】テキストエディタ使用者が条件翻訳変数ウィンドウ３０２に対して変更操作を行っていない場合、テキストエディタ使用者がソーステキスト１０６中の#define条件翻訳指示子の変更または新規追加を行ったか判定し（ステップ１９１２）、変更または新規追加を行った場合、ステップ１９０２に戻り、ソーステキスト１０６中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ１９０３で条件翻訳変数テーブルを再構築し、ステップ１９０４でソーステキスト１０６中の条件翻訳変数に影響されない共通ソーステキスト部分１と、翻訳条件が成立する条件翻訳領域をコンソール１１１に再表示し、ステップ１９０５で条件翻訳変数テーブル１０８の内容を条件翻訳変数ウィンドウ３０２に再表示し、以降のステップを繰り返す。

【０１０５】テキストエディタ使用者が#define条件翻訳指示子の変更または新規追加を行っていない場合、テキストエディタ使用者がソーステキスト１０６中の#if、#elseおよび#endifの各条件翻訳指示子を追加・変更・削除したかどうか判定する（ステップ１９１３）。各条件翻訳指示子を追加・変更・削除していない場合、ステップ１９０６に戻り以降のステップを繰り返す。

【０１０６】各条件翻訳指示子を追加・変更・削除した場合、ステップ１９０２に戻り、ソーステキスト１０６中の共通ソーステキスト部分１および共通ソーステキスト部分２と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ１９０３で条件翻訳変数テーブルを再構築し、ステップ１９０４でソーステキスト１０６中の条件翻訳変数に影響されない共通ソーステキスト部分１と、翻訳条件が成立する条件翻訳領域をコンソール１１１に再表示し、以降のステップを繰り返す。

【０１０７】図１８は、図１７におけるステップ１９０３で実行される、条件翻訳変数テーブル作成処理１１４の処理の流れを説明するフローチャートである。

【０１０８】図１８では、ソーステキスト１０６からソーステキストを１行づつ取り出し（ステップ６０１）、ソーステキスト終端か判定し（ステップ６０２）、ソーステキスト終端の場合、ソーステキスト１０６中に共通ソーステキスト部分１が存在するか判定し（ステップ６０３）、存在する場合、条件翻訳変数テーブル１０８に、共通ソーステキスト部分１を示す特別なエントリを追加し（ステップ６０４）、テキストエディタ使用者に共通ソーステキスト部分１として指定された背景色を追加した特別なエントリに設定する（ステップ６０５）。

次にソーステキスト１０６中に共通ソーステキスト部分２が存在するか判定し（ステップ６０６）、存在する場合、条件翻訳変数テーブル１０８に、共通ソーステキスト部分２を示す特別なエントリを追加し（ステップ６０７）、テキストエディタ使用者に共通ソーステキスト部分２として指定された背景色を追加した特別なエントリに設定し（ステップ６０８）、呼び出し元に処理を戻す。

【０１０９】ソーステキスト終端以外の場合、取り出したソーステキスト行の行頭が「#define」か判定し（ステップ６０９）、行頭が「#define」の場合、#define条件翻訳指示子から条件翻訳変数の名称と値を抽出し（ステップ６１０）、条件翻訳変数テーブル１０８に新しいエントリを追加し（ステップ６１１）、抽出した条件翻訳変数の名称と値と取り出した行の行番号と、この条件翻訳変数に影響されるソーステキスト部分としてテキストエディタ使用者に指定された背景色を、追加したエントリに設定し（ステップ６１２）、ステップ６０１に戻り以降の処理を繰り返す。

【０１１０】行頭が「#define」以外の場合、行頭が「#if」かどうか判定し（ステップ６１３）、行頭が「#if」以外の場合、ステップ６０１に戻り以降のステップを繰り返す。

【０１１１】行頭が「#if」の場合、#if条件翻訳指示子の翻訳条件中に指定された条件翻訳変数の名称を抽出し（ステップ６１４）、抽出した条件翻訳変数の名称が既に条件翻訳変数テーブル１０８に存在するか判定する（ステップ６１５）。既に条件翻訳変数テーブル１０８に存在する場合、ステップ６０１に戻り以降のステップを繰り返す。

【０１１２】抽出した条件翻訳変数の名称が条件翻訳変数テーブル１０８に存在しない場合、条件翻訳変数テーブル１０８に新しいエントリを追加し（ステップ６１６）、抽出した条件翻訳変数の名称と、この条件翻訳変数が未定義であることを示す値と、この条件翻訳変数に影響されるソーステキスト部分としてテキストエディタ使用者に指定された背景色を、追加したエントリに設定し（ステップ６１７）、ステップ６０１に戻り以降の処理を繰り返す。

【０１１３】以上で説明した実施例によって、本発明の第一の実施例と同様の効果を持ちながら、本発明の第一の実施例により表示したソーステキストに対して、テキストエディタ使用者が、条件翻訳変数を容易に変更する手段を提供することで、種々の翻訳条件の設定を可能にし、設定された翻訳条件に基づいて展開されたソーステキストを容易に確認できる。更に本実施例によれば、ソーステキスト中に分散して記述されている任意の条件翻訳変数に対する参照・変更が容易になるという効果を持つ。

【０１１４】なお、以上で説明した実施例では、ソーステキストウィンドウ３０１には、ソーステキスト中の条件翻訳指示子の翻訳条件を判定し、翻訳条件が成立する条件翻訳領域だけを表示するという表示方法であるが、本実施例における条件翻訳指示子の翻訳条件の判定および判定結果に基づいて条件翻訳領域の表示の有無を決定する処理の代わりに、ソーステキスト中の条件翻訳領域をアウトライン表示する処理を実行することによって、ソーステキストウィンドウ３０１に条件翻訳領域をアウトライン表示することができ、かつ条件翻訳変数ウィンドウ３０２に条件翻訳変数テーブル１０８の内容を表示し、テキストエディタ使用者はこのウィンドウに表示された条件翻訳変数テーブル１０８の内容を参照・変更するとともに、テキストエディタ使用者が条件翻訳変数テーブル１０８の内容を変更した場合、変更内容をソーステキスト１０６中の対応する#define条件翻訳指示子に反映し、条件翻訳指示子の翻訳条件の再判定を行い、ソーステキストウィンドウ３０１の表示内容を更新するようにできることは言うまでもない。この方法と本実施例で説明した方法を選択するように構成することによって、テキストエディタ使用者は、ソーステキストウィンドウ３０１の表示内容を、翻訳条件が成立する条件翻訳領域だけを表示するか、条件翻訳領域をアウトライン表示するか、必要に応じて切替えることができる。

【０１１５】以上で説明した本発明の第一、第二及び第三の実施例を組み合わせることにより、第一の実施例によるソーステキスト表示処理と、第二の実施例によるソーステキスト表示処理を切替えることで、テキストエディタ使用者は、ソーステキスト全体を表示させる方法と、条件翻訳領域において翻訳条件が成立する条件翻訳領域だけを表示させる方法と、翻訳条件が成立するか否かにかかわらず任意の条件翻訳領域を表示させる方法とを、必要に応じて任意に選択することができるとともに、各ソーステキスト表示方法において、条件翻訳変数を変更し、種々の翻訳条件の設定を可能にし、設定された翻訳条件に基づいて展開されたソーステキストを容易に確認できるという効果を持つ。

【０１１６】なお、以上で説明した本発明による第一、第二及び第三の実施例では表１の条件翻訳指示子を使用して説明したが、プログラム言語ＣＯＢＯＬの次期規格では、図１９のような条件翻訳指示子が規定される予定である。

【０１１７】本発明によるテキストエディタが図１９の条件翻訳指示子を読み込む場合、「>>WHEN」から次の「>>WHEN」までの領域を#if～#endifと置き換えてから本発明の第一、第二及び第三の実施例の処理と同様の処理を行うことで、本発明による第一、第二及び第三の実施例と同様の効果を持つことができる。

【０１１８】なお本発明の第一、第二及び第三の実施例を組み合わせることにより、テキストエディタ使用者は、ソーステキスト全体を表示させる方法と、条件翻訳

領域において翻訳条件が成立する条件翻訳領域だけを表示させる方法と、翻訳条件が成立するか否かにかかわらず任意の条件翻訳領域を表示させる方法とを、必要に応じて任意に選択することができるとともに、各ソーステキスト表示方法において、条件翻訳変数を変更し、種々の翻訳条件の設定を可能にし、設定された翻訳条件に基づいて展開されたソーステキストを容易に確認できるという効果を持つ。

【０１１９】

【発明の効果】本発明によれば、従来技術で問題であった、条件翻訳領域が入れ子構造になっているような複雑な条件翻訳を含むソーステキストの場合に、翻訳条件に従って展開されたソーステキストを効率よく表示することができないという問題を軽減することができる。また、従来技術で問題であった、条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中の条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できないため、共通なソーステキスト部分に対する修正ミスが発生しやすいという問題を軽減することもできる。

【図面の簡単な説明】

【図１】本発明の第一の実施形態によるテキストエディタの構成例を示すブロック図である。

【図２】従来技術の問題点の例を説明する図である。

【図３】本発明によるテキストエディタの説明で共通に使用する、入力ファイルの例である。

【図４】本発明の第一の実施形態によるテキストエディタが動作しているコンソールの画面表示例である。

【図５】本発明の第一の実施形態によるテキストエディタのソーステキストウィンドウの表示例である。

【図６】本発明の第一の実施形態によるテキストエディタの処理の流れを示すフローチャートである。

【図７】本発明の第一の実施形態によるテキストエディタの背景色設定処理の処理の流れを示すフローチャートである。

【図８】本発明の第一の実施形態によるテキストエディタの条件成立翻訳領域表示処理の処理の流れを示すフローチャートである。

【図９】本発明の第二の実施形態によるテキストエディタの構成例を示すブロック図である。

【図１０】本発明の第二の実施形態によるテキストエディタのソーステキストウィンドウの表示例である。

【図１１】本発明の第二の実施形態によるテキストエディタの、図１０における非展開状態のアウトライン表示領域に対して、展開するよう操作を行った場合の表示例である。

【図１２】本発明の第二の実施形態によるテキストエディタの処理の流れを示すフローチャートである。

【図１３】本発明の第二の実施形態によるテキストエディタのアウトライン表示処理の処理の流れを示すフローチャートである。

【図１４】本発明の第三の実施形態によるテキストエディタの構成例を示すブロック図である。

【図１５】本発明の第三の実施形態によるテキストエディタのコンソールの表示画面例である。

【図１６】本発明の第三の実施形態によるテキストエディタの条件翻訳変数ウィンドウの表示例である。

【図１７】本発明の第三の実施形態によるテキストエディタの処理の流れを示すフローチャートである。

【図１８】本発明の第三の実施形態によるテキストエディタの条件翻訳変数テーブル作成処理の処理の流れを示すフローチャートである。

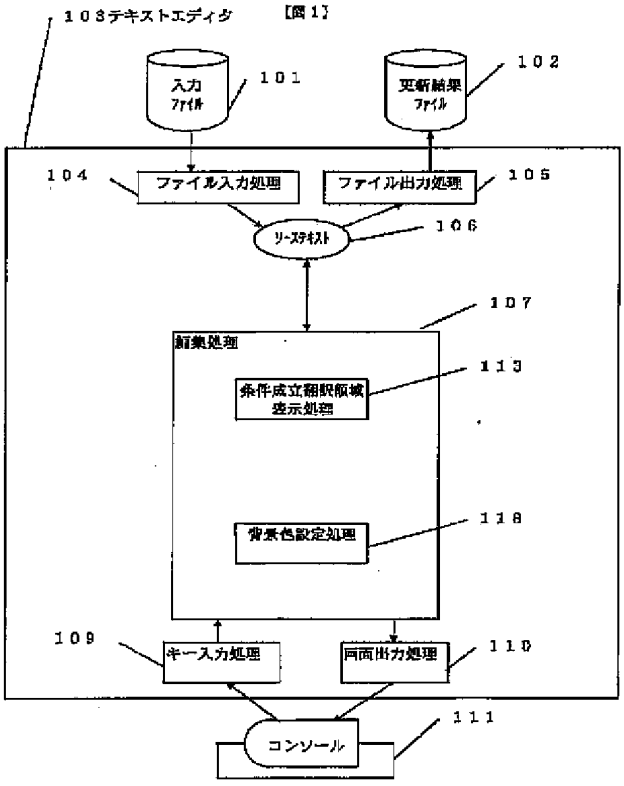
【図１９】プログラム言語ＣＯＢＯＬの次期規格における条件翻訳指示子の例である。

【図２０】条件翻訳指示子の文法と意味を示す。

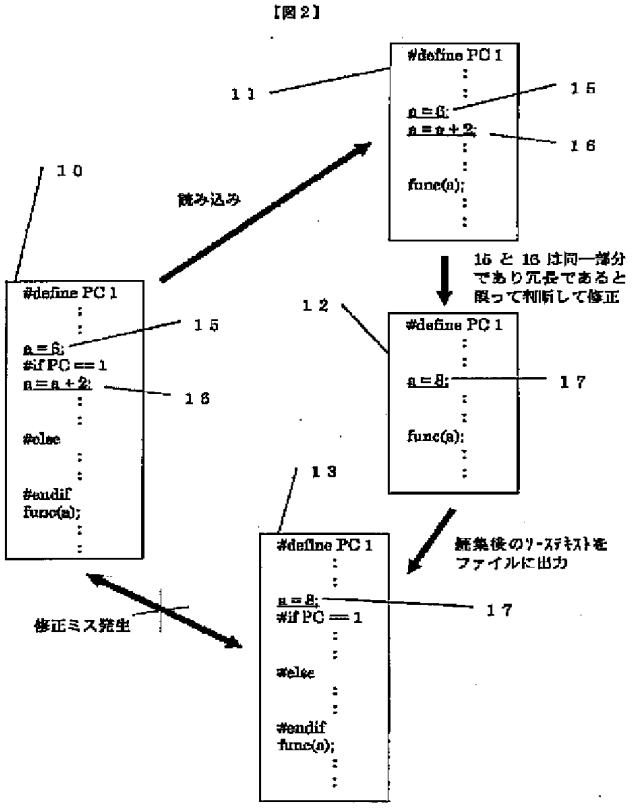
【符号の説明】

１０１　入力ファイル
１０２　更新結果ファイル
１０３　テキストエディタ
１０４　ファイル入力処理
１０５　ファイル出力処理
１０６　ソーステキスト
１０７　編集処理
１０８　条件翻訳変数テーブル
１０９　キー入力処理
１１０　画面出力処理
１１１　コンソール
１１３　条件成立翻訳領域表示処理
１１４　条件翻訳変数テーブル作成処理
１１５　条件翻訳変数テーブル表示制御処理
１１７　アウトライン表示処理
１１８　背景色設定処理

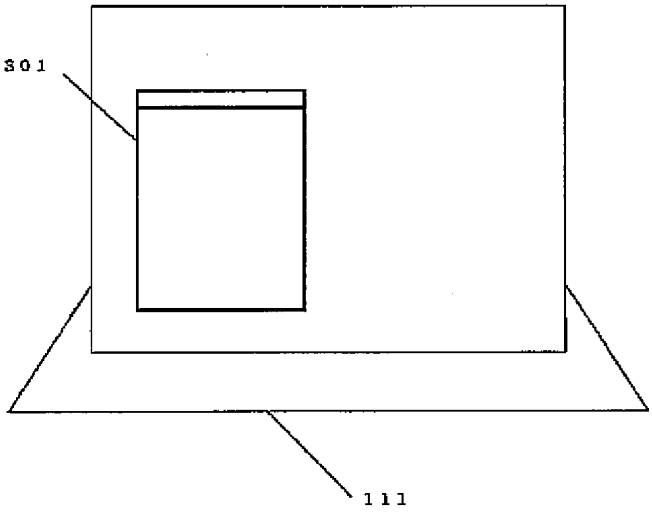
【図１】


【図１】
１０３テキストエディタ
入力ファイル
１０１
更新結果ファイル
１０２
１０４
ファイル入力処理
ファイル出力処理
１０５
ソーステキスト
１０６
１０７
編集処理
条件成立翻訳領域表示処理
１１３
背景色設定処理
１１８
１０９
キー入力処理
画面出力処理
１１０
コンソール
１１１

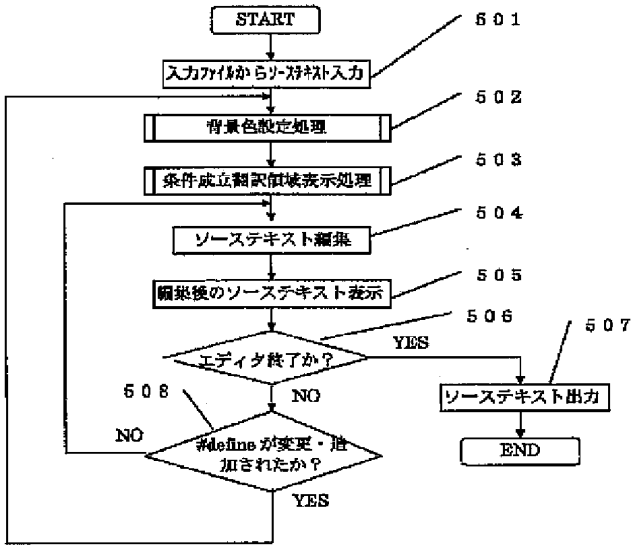
【図２】


【図２】
１１
１０
読み込み
１５
１６
１５ と １６ は同一部分であり冗長であると限って判断して修正
１２
１７
１３
編集後のソーステキストをファイルに出力
修正ミス発生
１７

【図４】


【図４】
３０１
１１１

【図６】


【図６】
START
５０１
入力ファイルからソーステキスト入力
５０２
背景色設定処理
５０３
条件成立翻訳領域表示処理
５０４
ソーステキスト編集
５０５
編集後のソーステキスト表示
５０６
エディタ終了か？
YES
５０７
ソーステキスト出力
END
NO
５０８
NO
#define が変更・追加されたか？
YES

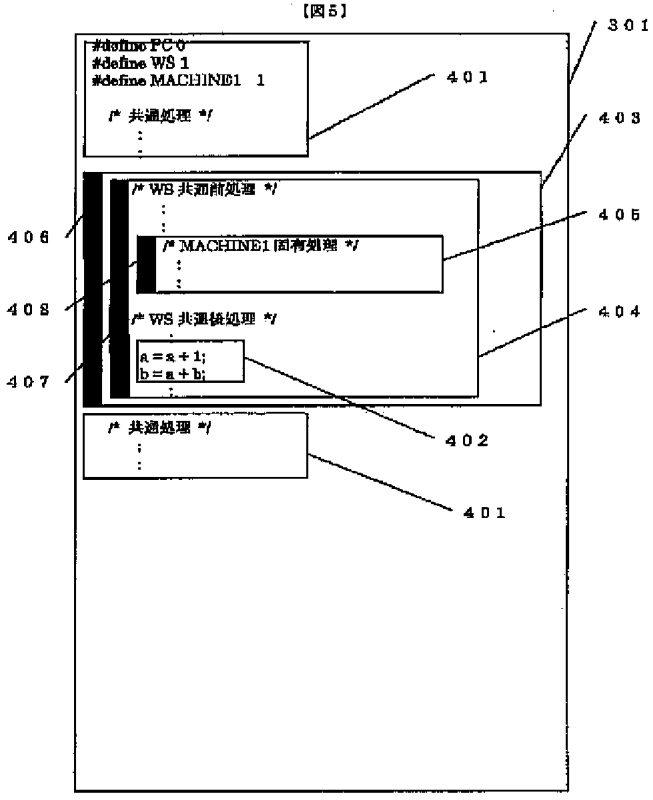
【図３】

【図３】

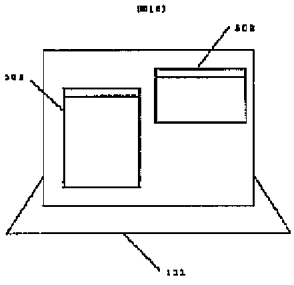
```
#define PC 0
#define WS 1
#define MACHINE1 1

/* 共通処理 */
   :
   :
#if PC == 1
/* PC 固有処理 */
   :
a = a + 1;
b = a + b;
   :
#else
#if WS == 1
/* WS 共通前処理 */
   :
   :
#if MACHINE1 == 1
/* MACHINE1 固有処理 */
   :
   :
#else
/* MACHINE1 以外の WS 固有処理 */
   :
   :
#endif
/* WS 共通後処理 */
   :
a = a + 1;
b = a + b;
   :
#else
/* PC/WS 以外の処理 */
   :
   :
#endif
#endif

/* 共通処理 */
   :
   :
```

【図５】

【図５】



【図１５】



【図１６】

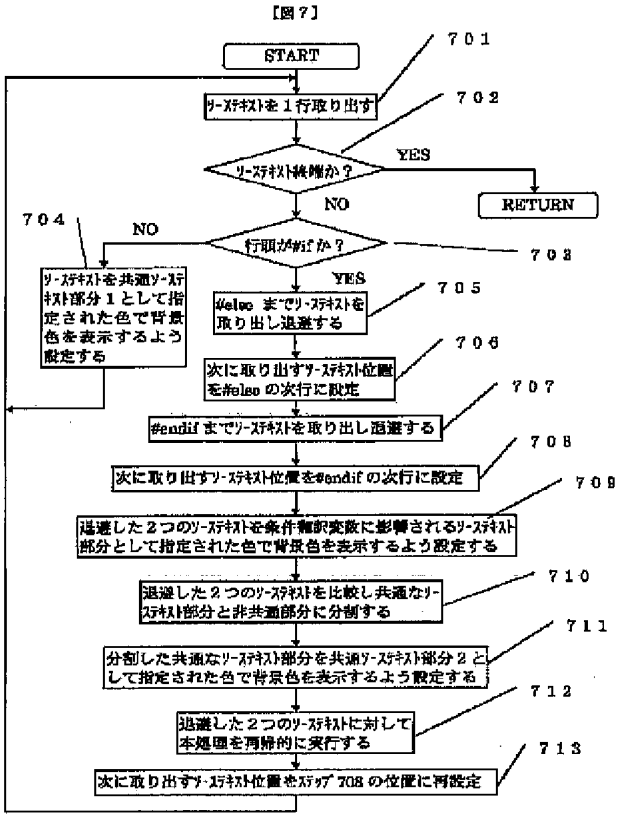
【図１６】

項番	名称	値	行番号	背景色
1	PC	0	1	赤
2	WS	1	2	緑
3	MACHINE1	1	3	青
4	共通ソースステキスト部分１	—	—	白
5	共通ソースステキスト部分２	—	—	黒

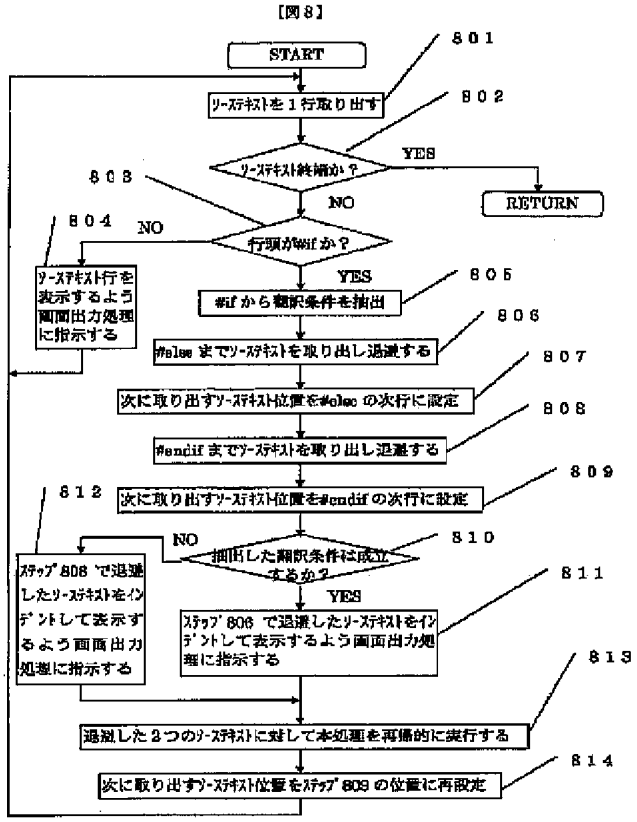
1801 1802 1803 1804 1805

302

【図７】



【図８】



【図１９】

【図１９】

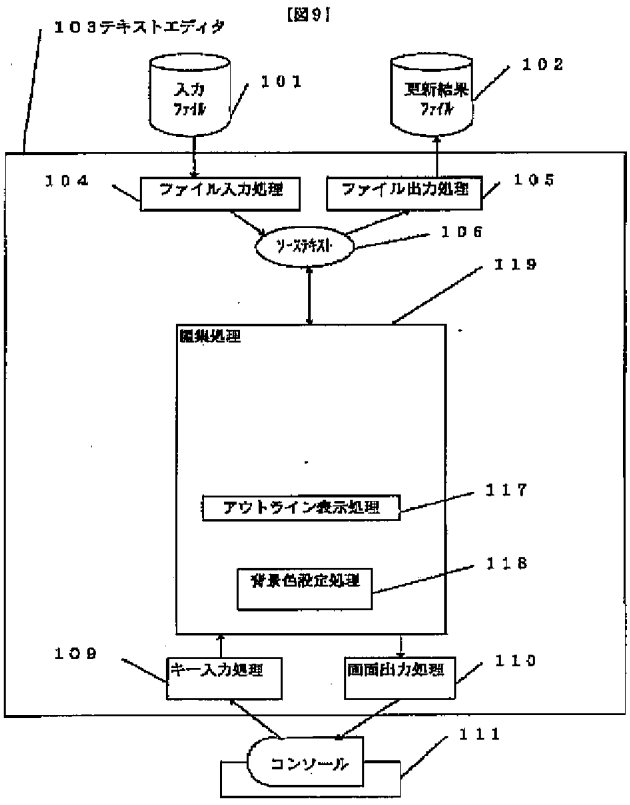
```
>>EVALUATE　翻訳条件
>>WHEN　翻訳条件
        実行文
          :
          :
>>WHEN　翻訳条件
        実行文
          :
          :
>>WHEN　OTHER
        実行文
          :
          :
>>END－EVALUATE
```

【図２０】

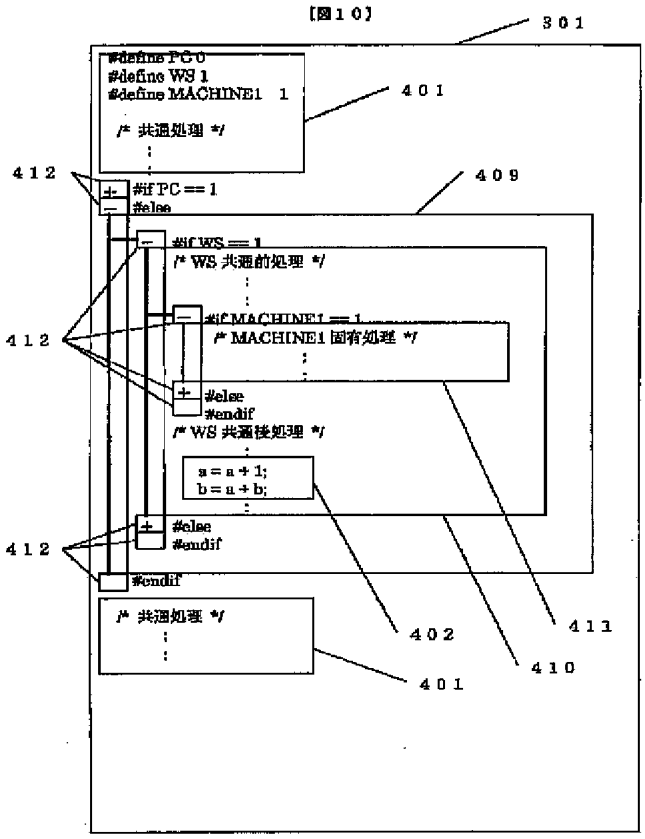
【図２０】

指示子	文法	説明
#define	#define 名称 値	#if 指示子の翻訳条件式で使用する条件翻訳変数を定義する。
#if	#if 翻訳条件式	翻訳条件式が成立する場合、対応する#else が存在すれば#else までの範囲のソーステキストを、対応する#else が存在しない場合は#endif までの範囲のソーステキストを読み込む。 翻訳条件式が成立しない場合、対応する#else が存在すれば#else 以降#endif までの範囲のソーステキストを読み込み、対応する#else が存在しない場合は#endif までの範囲のソーステキストを読み飛ばす。
#else	#else	対応する#if 指示子の翻訳条件式が成立しない場合に読み込むソーステキスト範囲の開始を示す。
#endif	#endif	条件翻訳のソーステキスト範囲の終了を示す。

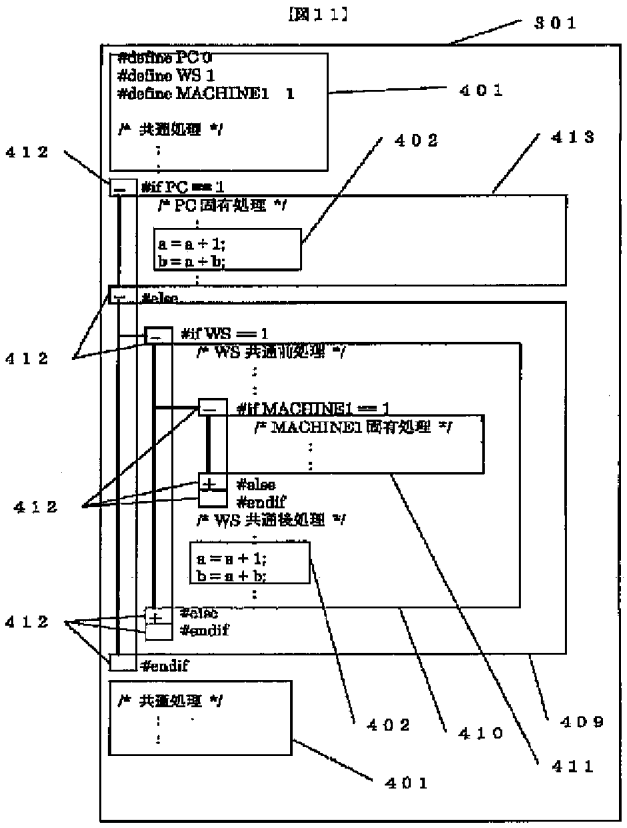
【図９】


【図9】
103テキストエディタ
入力ファイル
101
更新結果ファイル
102
104
ファイル入力処理
ファイル出力処理
105
ソーステキスト
106
119
編集処理
アウトライン表示処理
117
背景色設定処理
118
109
キー入力処理
画面出力処理
110
コンソール
111

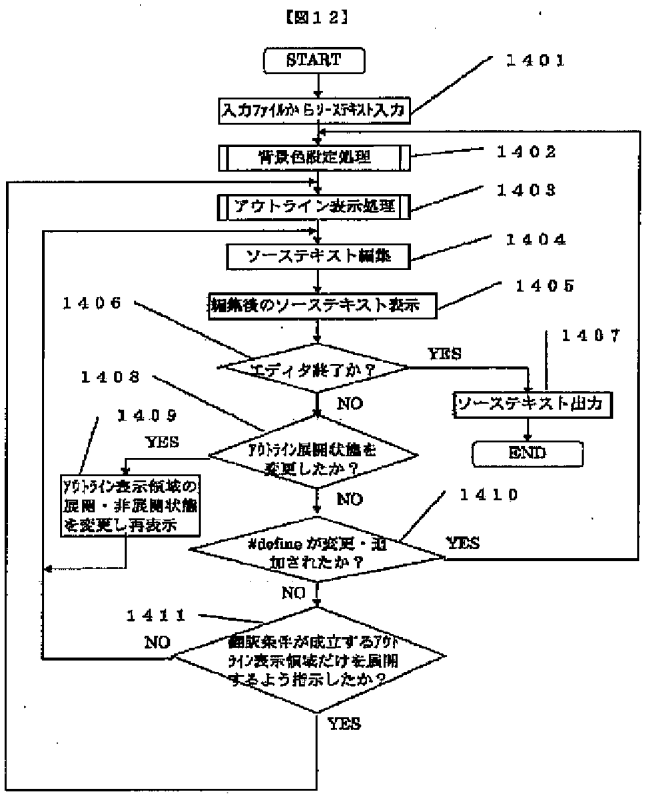
【図１０】


【図10】
301
#define PC 0
#define WS 1
#define MACHINE1 1
401
/* 共通処理 */
412
#if PC == 1
#else
409
#if WS == 1
/* WS 共通的処理 */
412
#if MACHINE1 == 1
/* MACHINE1 固有処理 */
#else
#endif
/* WS 共通後処理 */
a = a + 1;
b = a + b;
412
#else
#endif
#endif
/* 共通処理 */
402
411
410
401

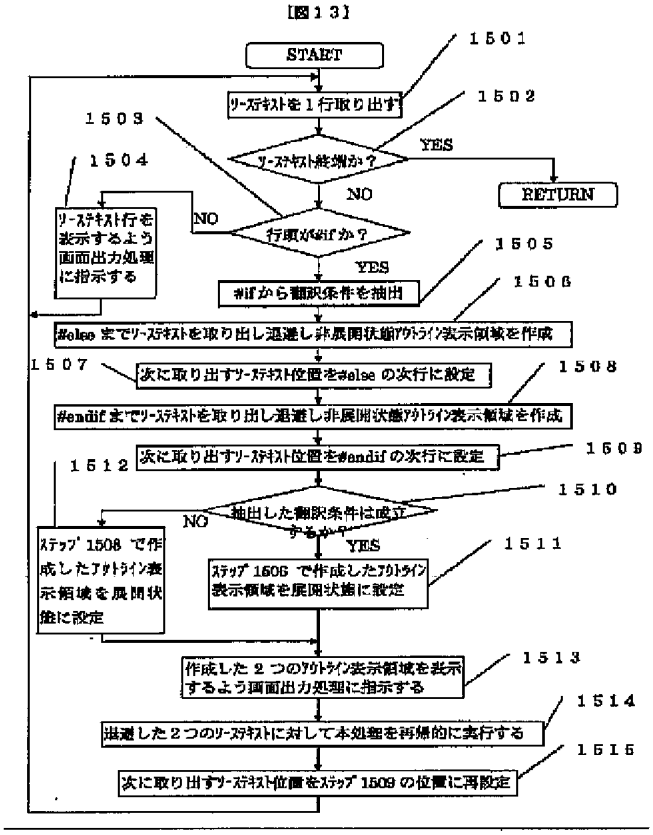
【図１１】


【図11】
301
#define PC 0
#define WS 1
#define MACHINE1 1
/* 共通処理 */
401
412
402
413
#if PC == 1
/* PC 固有処理 */
a = a + 1;
b = a + b;
#else
412
#if WS == 1
/* WS 共通前処理 */
#if MACHINE1 == 1
/* MACHINE1 固有処理 */
412
#else
#endif
/* WS 共通後処理 */
a = a + 1;
b = a + b;
412
#else
#endif
#endif
/* 共通処理 */
402
410
409
411
401

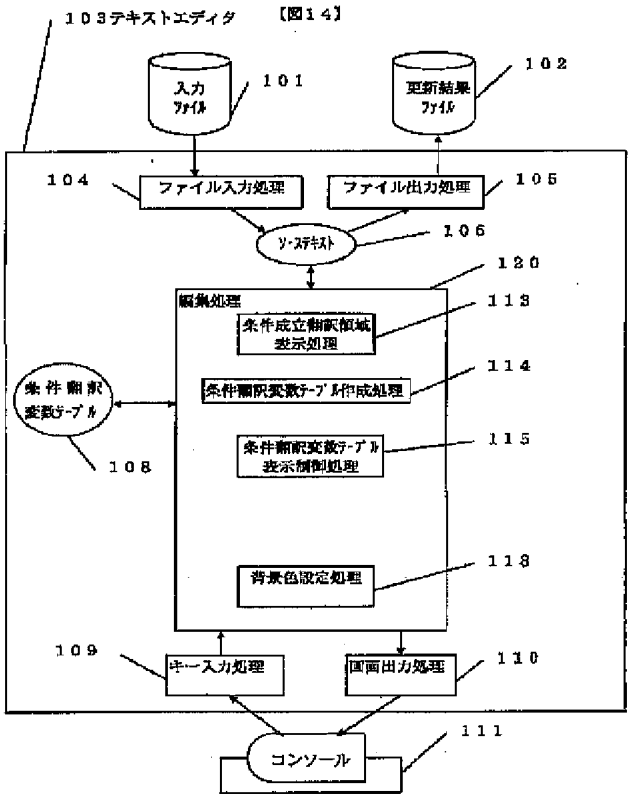
【図１２】


【図12】
START
1401
入力ファイルからソーステキスト入力
背景色設定処理
1402
アウトライン表示処理
1403
ソーステキスト編集
1404
編集後のソーステキスト表示
1405
1406
エディタ終了か？
YES
1407
NO
ソーステキスト出力
END
1408
1409
YES
アウトライン展開状態を変更したか？
アウトライン表示領域の展開・非展開状態を変更し再表示
NO
1410
#define が変更・追加されたか？
YES
NO
1411
NO
画面条件が成立するアウトライン表示領域だけを展開するよう指示したか？
YES

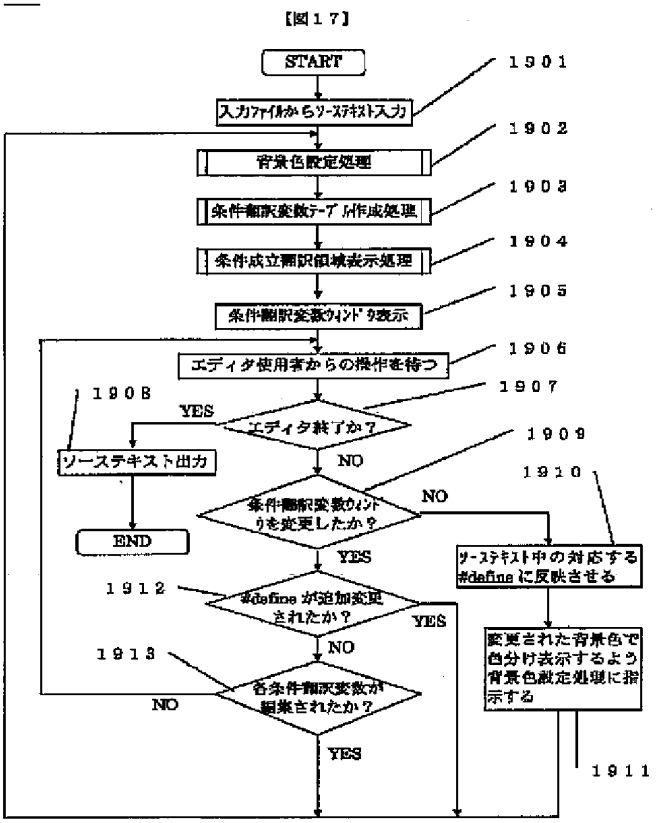
【図１３】


【図１３】
START
1501
ソーステキストを１行取り出す
1502
ソーステキスト終端か？
YES
RETURN
NO
1503
行頭が#if か？
NO
1504
ソーステキスト行を表示するよう画面出力処理に指示する
YES
1505
#if から翻訳条件を抽出
1506
#else までソーステキストを取り出し退避し非展開状態アウトライン表示領域を作成
1507
次に取り出すソーステキスト位置を#else の次行に設定
1508
#endif までソーステキストを取り出し退避し非展開状態アウトライン表示領域を作成
1509
次に取り出すソーステキスト位置を#endif の次行に設定
1510
抽出した翻訳条件は成立するか？
NO
YES
1511
ステップ 1506 で作成したアウトライン表示領域を展開状態に設定
1512
ステップ 1508 で作成したアウトライン表示領域を展開状態に設定
1513
作成した２つのアウトライン表示領域を表示するよう画面出力処理に指示する
1514
退避した２つのソーステキストに対して本処理を再帰的に実行する
1515
次に取り出すソーステキスト位置をステップ 1509 の位置に再設定

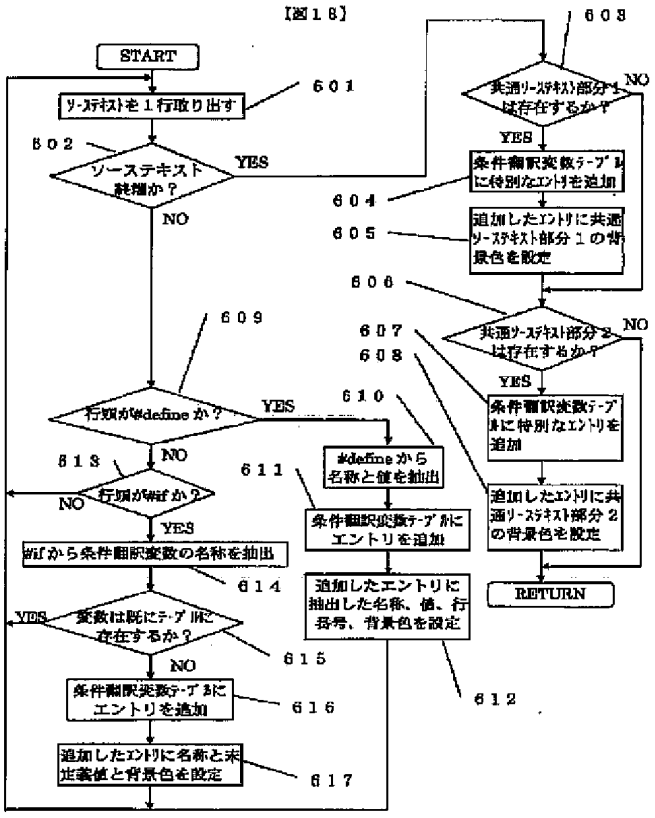
【図１４】


【図１４】
１０３テキストエディタ
入力ファイル
１０１
更新結果ファイル
１０２
１０４
ファイル入力処理
ファイル出力処理
１０５
ソーステキスト
１０６
１２０
編集処理
条件成立翻訳領域表示処理
１１３
条件翻訳変数テーブル
１１４
条件翻訳変数テーブル作成処理
条件翻訳変数テーブル表示制御処理
１１５
１０８
背景色設定処理
１１８
１０９
キー入力処理
画面出力処理
１１０
コンソール
１１１

【図１７】


【図１７】
START
1901
入力ファイルからソーステキスト入力
1902
背景色設定処理
1903
条件翻訳変数テーブル作成処理
1904
条件成立翻訳領域表示処理
1905
条件翻訳変数ウィンドウ表示
1906
エディタ使用者からの操作を待つ
1907
エディタ終了か？
YES
1908
NO
ソーステキスト出力
END
1909
条件翻訳変数ウィンドウを変更したか？
NO
1910
YES
ソーステキスト中の対応する#defineに反映させる
1912
#defineが追加変更されたか？
YES
NO
変更された背景色で色分け表示するよう背景色設定処理に指示する
1913
各条件翻訳変数が編集されたか？
NO
YES
1911

【図１８】


【図１８】
START
ソーステキストを１行取り出す
601
602
ソーステキスト終端か？
YES
NO
603
共通ソーステキスト部分１は存在するか？
NO
YES
条件翻訳変数テーブルに特別なエントリを追加
604
追加したエントリに共通ソーステキスト部分１の背景色を設定
605
606
609
607
608
共通ソーステキスト部分２は存在するか？
NO
YES
条件翻訳変数テーブルに特別なエントリを追加
追加したエントリに共通ソーステキスト部分２の背景色を設定
RETURN
行頭が#defineか？
YES
NO
610
#defineから名称と値を抽出
611
条件翻訳変数テーブルにエントリを追加
追加したエントリに抽出した名称、値、行番号、背景色を設定
612
613
行頭が#ifか？
NO
YES
#ifから条件翻訳変数の名称を抽出
614
変数は既にテーブルに存在するか？
YES
NO
615
条件翻訳変数テーブルにエントリを追加
616
追加したエントリに名称と未定義値と背景色を設定
617